A practical mixed-integer programming model for the vertex separation number problem

Sven Mallach

Institut für Informatik
Universität zu Köln, 50969 Köln, Germany

October 16, 2015

Abstract

We present a novel mixed-integer programming formulation for the vertex separation number problem in general directed graphs. The model is conceptually simple and, to the best of our knowledge, much more compact than existing ones. First experiments give hope that it can solve larger instances than has been possible so far if it is combined with pre-processing techniques to reduce the search space.

1 Introduction

Let $G = (V, E)$ ($G = (V, A)$) be an undirected (directed) graph and let $\Pi(V)$ be the set of all possible permutations of the vertices V of G.

For a given permutation $\pi \in \Pi(V)$, denote with $\pi(v)$ the position of each $v \in V$ in π. Suppose now that the vertices V are put on a line in the order specified by π. We say that π defines this way a *linear ordering* of V.

With this illustration in mind, it is easy to define the following two sets associated with each $v \in V$ and a given permutation $\pi \in \Pi(V)$:

$$L(\pi, v) = \{u \in V \mid \pi(u) < \pi(v)\}$$
$$R(\pi, v) = \{w \in V \mid \pi(v) \leq \pi(w)\}$$

The sets $L(\pi, v)$ and $R(\pi, v)$ can be thought of being generated by a cut through the linear ordering that is carried out marginally close to the left of v as is illustrated in Fig. 2 for vertex 4 and the ordering $\pi = \langle 1, 2, 3, 4, 5, 6 \rangle$.

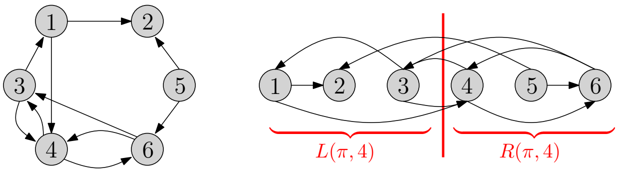

Figure 1: A linearly ordered directed graph and a cut illustration.

For an undirected graph $G = (V, E)$ or a directed graph $G = (V, A)$, the respective vertex separation associated to the ordering $\pi \in \Pi(V)$ is

$$vsn(\pi, G) = \max_{v \in V} |\{u \in L(\pi, v) \mid \exists w \in R(\pi, v) \text{ s.t. } \{u, w\} \in E\}|$$

and

$$vsn(\pi, G) = \max_{v \in V} |\{u \in L(\pi, v) \mid \exists w \in R(\pi, v) \text{ s.t. } (w, u) \in A\}|.$$

In the following, we will mostly deal with the directed case like in our example. For any fixed ordering $\pi \in \Pi(V)$ and each $v \in V$, we can imagine to write the corresponding values $|\{u \in L(\pi, v) \mid \exists w \in R(\pi, v) \text{ s.t. } (w, u) \in A\}|$ directly below the cutting line that belongs to v, together with the set of vertices in the set $L(\pi, v)$ that are hit by arcs (edges) coming from $R(\pi, v)$. This is illustrated in Fig. 2 for the example graph and ordering from Fig. 1.

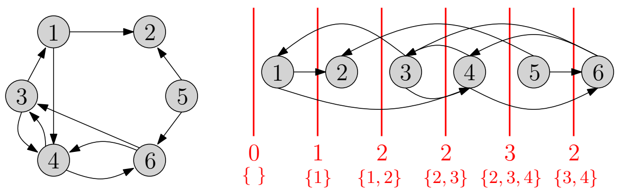

Figure 2: Values and sets associated to a given ordering.

The vertex separation number problem is now to find an ordering $\pi^* \in \Pi(V)$, such that $vsn(\pi^*, G)$ is minimum, i.e., to find

$$vsn(G) = \min_{\pi \in \Pi(V)} \max_{v \in V} |\{u \in L(\pi, v) \mid \exists w \in R(\pi, v) \text{ s.t. } (w, u) \in A\}|.$$

Fig. 3 shows an ordering that leads to the optimal vertex separation number $vsn(G) = 1$ for the example graph from Fig. 1.

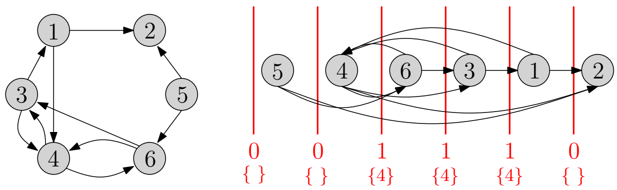

Figure 3: A linear ordering that yields the optimum vertex separation number.

As will be discussed in the following, practical exact approaches to the vertex separation number problem are rather rare. Especially, existing integer programming formulations are often very large in their size such that their successful application for larger instances with more than about 20 vertices is unlikely. The vertex separation number problem has often been explained in a linear ordering context (just as before), but to the best of our knowledge (and to our surprise), there is no solution approach so far that is based on *linear ordering variables.* In this paper, we therefore propose such a new formulation that is considerably smaller in size as most of the existing approaches and also conceptually simple.

2 Related Work

Deciding whether an arbitrary directed or undirected graph has vertex separation number at most k is $\mathcal{NP}$-complete [10]. It also remains $\mathcal{NP}$-complete for several more restricted graph classes [5]. In the following, we restrict ourselves to research that deals with the (exact) solution of the vertex separation number problem or the pathwidth problem which are proven to be equivalent in [9]. For an overview of the several other (equivalence) relations between the vertex separation number and other graphtheoretical problems as well as applications, we refer to [5].

As before, we consider undirected graphs $G = (V, E)$, and directed graphs $G = (V, A)$, such that $n = |V|$ and $m = |E|$ or $m = |A|$.

Duarte et al. [6] give both, an IP formulation and a variable neighborhood search algorithm, for the case of undirected graphs. Further, they introduced the benchmark library VSPLIB that consists of 173 instances in total and can be divided into the following subsets:

- HB (directed, symmetric): 73 instances, $24 \leq n \leq 960$, $34 \leq m \leq 3721$.
- Grids (undirected): 50 instances, $25 \leq n \leq 2916$, $40 \leq m \leq 5724$.
- Trees (undirected): 50 instances, $22 \leq n \leq 202$, $21 \leq m \leq 201$.

The proposed IP model has $\mathcal{O}(mn^2)$ variables and $\mathcal{O}(mn^3)$ constraints. Due to the large size, the authors could apply their model only for 23 VSPLIB-instances. In their experiments, only one of these 23 instances could could be solved within 90 minutes computation time.

Another IP formulation was given by Solano and Pióro [12] who studied the problem in the (directed) context of *wavelength-division multiplexing* in optical communication networks. This model is, to the best of our knowledge, the only one that is at least asymptotically of a size as small as the one we are going to present in this paper. It has $3n^2 + 1$ variables in total, but only n of them need an explicitly enforced (binary) integrality property. Further, the number of constraints is only $\mathcal{O}(mn)$. Computational experience with this model is hardly published, the authors only report that a CPLEX implementation of their model took two hours in order to solve a particular 20-vertex instance. However, an adapted and therefore very similar model has been implemented in the mathematics library Sage [7]. Computational results in [3, 4] and own experiments show that this formulation is unlikely to solve instances with more than 30 vertices in reasonable time as well, because the root node relaxation is already too large. However, for small instances it performs similarly to the approach we present here. For that reason, we briefly discuss the model in Sect. 3. A further dynamic programming algorithm implemented in Sage is restricted to instances with at most 31 vertices.

Since the performance of the MIP model was not satisfactory for their application, Solano and Pióro also developed a combinatorial branch-and-bound algorithm that was then subject to extensions and improvements by Coudert et al. [3]. A comprehensive experimental study is given in [4] that also includes benchmarks using the VSPLIB. Due to several search space reductions, Coudert et al.'s algorithm appears to be very fast (< 1s) on many instances, but still it cannot solve the majority of the instances within 10 minutes (which is very challenging).

Gurski [8] provides another IP formulation for the vertex separation problem for undirected graphs that can also be modified to deal with directed graphs. Again, the basic idea is to create the right linear layout for the vertices, but without using linear ordering variables. The model has such a large size ($n^4 + n^2 + 1$ variables, mainly due to linearizations) that it cannot be applied to larger instances in practice.

A further IP formulation for undirected graphs has been given by Biedl et al [2]. An implementation of that model was told to take several minutes for instances with less than 10 vertices. It was therefore transformed into a SAT model where graphs with $n + m < 45$ could almost always be solved, but those where $n + m > 70$ could almost never be solved.

Finally, a more theoretically motivated enumerative algorithm whose running time can be bounded by $\mathcal{O}(1.9657^n)$ is presented by Suchan and Villanger in [13].

3 Set-based Ordering Model

For comprehensiveness, we state the IP model that has been implemented into Sage [7] and is a modification of the original formulation by Solano and Pióro [12]. It computes an ordering of the vertices V by enforcing a collection of sets $\mathcal{S} = \{S_1, \ldots, S_{|V|}\}$ such that $|S_i| = i$, and $S_i \subset S_j$ for any $i < j$. That is, S_1 specifies the vertex that is ranked first, and in general, the rank $\pi(v)$ of vertex $v \in V$ is $\pi(v) = \min\{i \mid v \in S_i\}$.

As opposed to the definition of the vertex separation problem for directed graphs presented so far, the model implemented into Sage counts and minimizes over all vertices $w \in V$ the maximum number of vertices $w' \in R(\pi, w)$ such that there exists a vertex $v \in L(\pi, w)$ with $(v, w') \in A$. This is equivalent since the roles of L and R can be exchanged if one exchanges the measured arc directions as well. So any vertex ordering obtained this way just needs to be reversed in order to yield a solution of the same quality conforming to our definition.

We first state the model in its original form. It has the following variables:

- $y_{v,i} = 1$ if $v \in S_i$ and $y_{v,i} = 0$ otherwise.
- $x_{v,i} = 1$ if there is a vertex $u \in S_i$ such that $(u, v) \in A$, and $x_{v,i} = 0$ otherwise.
- $u_{v,i} = 1$ if $v \notin S_i$ and there is a vertex $u \in S_i$ such that $(u, v) \in A$, and $u_{v,i} = 0$ otherwise.
- T: The objective variable that captures the vertex separation number.

As already indicated above, the smallest i such that $y_{v,i} = 1$ corresponds to the order of v in the linear layout interpretation. The full model is then:

$$
\begin{array}{llll}
\min & T & & \\
\text{s.t.} & x_{v,i} - x_{v,i+1} & \leq 0 & \text{for all } v \in V,\ 1 \leq i \leq |V|-1 \quad (1)\\
& y_{v,i} - y_{v,i+1} & \leq 0 & \text{for all } v \in V,\ 1 \leq i \leq |V|-1 \quad (2)\\
& y_{v,i} - x_{w,i} & \leq 0 & \text{for all } v, w \in V,\ (v,w) \in A,\ 1 \leq i \leq |V| \quad (3)\\
& \sum_{v \in V} y_{v,i} & = i & \text{for all } 1 \leq i \leq |V| \quad (4)\\
& x_{v,i} - y_{v,i} & \leq u_{v,i} & \text{for all } v \in V,\ 1 \leq i \leq |V| \quad (5)\\
& \sum_{v \in V} u_{v,i} & \leq T & \text{for all } 1 \leq i \leq |V| \quad (6)\\
& y_{v,i} & \in \{0,1\} & \text{for all } v \in V,\ 1 \leq i \leq |V| \quad (7)\\
& u_{v,i} & \geq 0 & \text{for all } v \in V,\ 1 \leq i \leq |V| \quad (8)\\
& u_{v,i} & \leq 1 & \text{for all } v \in V,\ 1 \leq i \leq |V| \quad (9)\\
& x_{v,i} & \geq 0 & \text{for all } v \in V,\ 1 \leq i \leq |V| \quad (10)\\
& x_{v,i} & \leq 1 & \text{for all } v \in V,\ 1 \leq i \leq |V| \quad (11)\\
& T & \geq 0 & \quad (12)
\end{array}
$$

The first two constraints are basically *forwarding constraints* that ensure that if $y_{v,i} = 1$ ($x_{v,i} = 1$) then also $y_{v,i+1} = 1$ ($x_{v,i+1} = 1$). Because, by definition, $v \in S_j$ for any $j > i$ if $v \in S_i$. And clearly, if v is hit by a vertex $u \in S_i$, then v is also hit by a vertex from any S_j, $j > i$, simply since $u \in S_j$ as well. Inequalities (3) make sure that $x_{w,i} = 1$ whenever $y_{v,i} = 1$ and $(v,w) \in A$, as stated in the above definition of the x-variables. Equations (4) enforce the correct cardinality of the sets S_i defined by the variables $y_{v,i}$. Inequalities (5) make sure that $u_{v,i} = 1$ whenever $x_{v,i} = 1$ and $y_{v,i} = 0$, i.e., if v is not in S_i but there is an edge $(u,v) \in A$ with $u \in S_i$ (v must be counted at the cut position i). Finally, the value of T is given by the maximum sum $\sum_{v \in V} u_{v,i}$ over all i.

To obtain a model that adheres to our reverse problem definition, we could reinterprete variables as:

- $x_{v,i} = 1$ if there is a vertex $w \notin S_i$ such that $(w,v) \in A$, and $x_{v,i} = 0$ otherwise.
- $u_{v,i} = 1$ if $v \in S_i$ and there is a vertex $w \notin S_i$ such that $(w,v) \in A$, and $u_{v,i} = 0$ otherwise.

Further, we need to change

- inequality (1) into $x_{v,i+1} - x_{v,i} \leq 0$,
- inequality (3) into $-y_{w,i} - x_{v,i} \leq -1$, and
- inequality (5) into $x_{v,i} + y_{v,i} - u_{v,i} \leq 1$.

4 A Novel Linear-Ordering Model

As already indicated, our new model is based on *linear ordering variables* $x_{i,j} \in \{0,1\}$ for each pair of vertices $i, j \in V$ such that $i < j$. The variable

$x_{i,j}$ is equal to 1 if and only if i precedes j in the linear ordering (and hence equal to 0 if j precedes i).

It is well known that integer value assignments to the linear ordering variables are in one-to-one correspondence with permutations of V if and only if they satisfy the so-called *3-dicycle-inequalities* [11]:

$$\begin{aligned} x_{i,j} + x_{j,k} - x_{i,k} &\geq 0 && \text{for all } i,j,k \in V, i < j < k \\ x_{i,j} + x_{j,k} - x_{i,k} &\leq 1 && \text{for all } i,j,k \in V, i < j < k \end{aligned}$$

For ease of reference, for a fixed ordering $\pi \in \Pi(V)$, let

$$S_v(\pi) = \{u \in L(\pi, v) \mid \exists w \in R(\pi, v) \text{ s.t. } (w, u) \in A\}$$

for each $v \in V$. The general idea is to have, for each $v \in V$, a variable $z_v \in \mathbb{Z}_{\geq 0}$ such that $z_v = |S_v(\pi)|$ holds for the ordering π that is expressed in the linear ordering variables. We achieve this using the equation

$$z_v = \sum_{u \in V,\, u \neq v} y_{u,v}$$

where the $y_{u,v}$ are binary variables stating whether u must be counted in the above expression for v ($y_{u,v} = 1$) or not ($y_{u,v} = 0$). It remains to explain how we enforce the correct values on the y-variables. Here, we may first exploit one nice property in the case that $(v, u) \in A$ and then restrict us separately to the case where $(v, u) \notin A$.

Lemma 4.1. *Let $u, v \in V$, $v \neq u$, and $(v, u) \in A$. Then $u \in S_v(\pi)$ if and only if u precedes v in π.*

Proof. First, suppose that u precedes v in π. Then clearly $u \in L(\pi, v)$. Since also $v \in R(\pi, v)$, the arc (v, u) satisfies the required existence property in the definition of $S_v(\pi)$ and hence $u \in S_v(\pi)$. Conversely, if $u \in S_v(\pi)$, then $u \in L(\pi, v)$ and therefore u must precede v in π. □

Lemma 4.2. *Let $u, v \in V$, $v \neq u$, and $(v, u) \notin A$. Then $u \in S_v(\pi)$ if and only if u precedes v in π and there is some $w \neq u, v$ such that w succeeds v in π and $(w, u) \in A$.*

Proof. The $\Leftarrow$-part follows directly from the definition of $S_v(\pi)$. For the $\Rightarrow$-part: Suppose $u \in S_v(\pi)$. Then $u \in L(\pi, v)$ and hence u must precede v in π. Assume now that there is no such w as required in the lemma. Then the only vertex in $R(\pi, v)$ that could cause $u \in S_v(\pi)$ is v itself. However, by assumption, $(v, u) \notin A$ which is a contradiction. □

Corollary 4.1. *If $(v, u) \in A$, then $y_{u,v} = 1$ if and only if $x_{u,v} = 1$, i.e., $x_{u,v} = y_{u,v}$.*

Corollary 4.2. *If $(v, u) \notin A$, then $y_{u,v} = 1$ if and only if $x_{u,v} = 1$ and there is a vertex $w \neq u, v$ such that $(w, u) \in A$ and $x_{v,w} = 1$. Hence, it holds that $y_{u,v} \geq x_{u,v} + x_{v,w} - 1$.*

So the full model is:

$$
\begin{array}{llll}
\min & Z & & \\
\text{s.t.} & x_{i,j} + x_{j,k} - x_{i,k} \geq 0 & \text{for all } i,j,k \in V, i<j<k & \\
 & x_{i,j} + x_{j,k} - x_{i,k} \leq 1 & \text{for all } i,j,k \in V, i<j<k & \\
 & x_{i,j} = y_{i,j} & \text{for all } i,j \in V, (j,i) \in A & (13) \\
 & x_{i,j} + x_{j,k} - 1 \leq y_{i,j} & \text{for all } i,j \in V, (j,i) \notin A, (k,i) \in A, j \neq i,k & (14) \\
 & \sum_{i \in V, i \neq j} y_{i,j} = z_j & \text{for all } j \in V & (15) \\
 & z_j \leq Z & \text{for all } j \in V & (16) \\
 & x_{i,j} \in \{0,1\} & \text{for all } i,j \in V, i<j & \\
 & y_{i,j} \geq 0 & \text{for all } i,j \in V, i \neq j & \\
 & y_{i,j} \leq 1 & \text{for all } i,j \in V, i \neq j & \\
 & z_j \geq 0 & \text{for all } j \in V & \\
 & Z \geq 0 & &
\end{array}
$$

This model has $\binom{n}{2} + n(n-1) + n + 1 = \mathcal{O}(n^2)$ variables. The n z-variables and the according equations (15) can be saved by directly plugging the relation into inequalities (16). Further, several y-variables are fixed to x-variables and can hence be omitted. The number of constraints is bounded by $\mathcal{O}(n^3)$.

Remark 4.1. *Since $x_{u,v} = 1$ is a necessary condition for $y_{u,v} = 1$, it also holds that $y_{u,v} \leq x_{u,v}$. These inequalities are however not necessary for the correctness of the model. Whenever the value of $y_{u,v}$ is implied by an equation (13) or $y_{u,v} = 1$ is implied by an inequality of type (14), integral x-values imply integral y-values. In any other case, $y_{u,v}$ can be set to* 0 *(which will be done as soon as this is relevant for the objective function). Hence, Z will be integral as well if all the x-values are, and reflect the correct associated objective function value.*

The 3-dicycle-inequalities are a natural candidate to be considered as cutting planes (i.e., they are not added to the initial LP, but separated instead). This enlarges the applicability to larger instances. In addition, every further inequality that is valid for the linear ordering problem (plenty exist, see e.g. [11]) is also valid for this formulation.

5 Outlook

We implemented both, the set-oriented and the novel model, using CPLEX 12.6. [1]. On a Debian Linux machine with an Intel Core i7-3770T processor running at 2.5 GHz and with 32 GB RAM, and with a time limit of 600 seconds (single-threaded runs), our model could solve 6 of the directed and 3 of the undirected VSPLIB instances. The set-oriented model could solve 8 and 2 instances, respectively. All the solved instances where in the range of about 25 (directed) and up to 120 (undirected) vertices. However, in 57 (directed) and 54 (undirected) cases, not even the root LP relaxation of the set-oriented model could be solved within the time limit. This was the case only once for our new model which needs only about half of the variables, even without taking into account further possibilities to set or omit variables. At the same time, our model provides good opportunities to start with only a subset of the constraints and to apply a cutting plane approach then. As this is a promising setting, we plan to combine our model with the search space reduction methods from [3, 4] and further investigate its practical applicability and performance.

References

[1] CPLEX optimization studio 12.6. Reference manual, IBM ILOG, 2013.

[2] Therese Biedl, Thomas Bläsius, Benjamin Niedermann, Martin Nöllenburg, Roman Prutkin, and Ignaz Rutter. Using ILP/SAT to determine pathwidth, visibility representations, and other grid-based graph drawings. In Stephan Wismath and Alexander Wolff, editors, *Graph Drawing*, volume 8242 of *Lecture Notes in Computer Science*, pages 460–471. Springer, 2013.

[3] David Coudert, Dorian Mazauric, and Nicolas Nisse. Experimental evaluation of a branch and bound algorithm for computing pathwidth. In Joachim Gudmundsson and Jyrki Katajainen, editors, *Experimental Algorithms*, volume 8504 of *Lecture Notes in Computer Science*, pages 46–58. Springer, 2014.

[4] David Coudert, Dorian Mazauric, and Nicolas Nisse. Experimental evaluation of a branch and bound algorithm for computing pathwidth. Research Report RR-8470, INRIA, Feb 2014.

[5] Josep Díaz, Jordi Petit, and Maria Serna. A survey of graph layout problems. *ACM Comput. Surv.*, 34(3):313–356, Sep 2002.

[6] Abraham Duarte, Laureano F. Escudero, Rafael Martí, Nenad Mladenovic, Juan José Pantrigo, and Jesús Sánchez-Oro. Variable neighborhood search for the vertex separation problem. *Computers & Operations Research*, 39(12):3247–3255, 2012.

[7] Michael Gray. Sage: A new mathematics software system. *Computing in Science & Engineering*, 10(6):72–75, 2008.

[8] Frank Gurski. Linear programming formulations for computing graph layout parameters. *The Computer Journal*, 2015.

[9] Nancy G. Kinnersley. The vertex separation number of a graph equals its path-width. *Inf. Process. Lett.*, 42(6):345–350, Jul 1992.

[10] Thomas Lengauer. Black-white pebbles and graph separation. *Acta Informatica*, 16(4):465–475, 1981.

[11] Rafael Martí and Gerhard Reinelt. *The Linear Ordering Problem.* Springer, 2011.

[12] Fernando Solano and Michal Pióro. Lightpath reconfiguration in WDM networks. *IEEE/OSA Journal of Optical Communications and Networking*, 2(12):1010–1021, December 2010.

[13] Karol Suchan and Yngve Villanger. Computing pathwidth faster than 2^n. In Jianer Chen and Fedor V. Fomin, editors, *Parameterized and Exact Computation*, volume 5917 of *Lecture Notes in Computer Science*, pages 324–335. Springer, 2009.